Low Complex-Hierarchical Coding Compression Approach for Arial Images

[1] M. Suresh, [2] J. Amulya Kumar
[1] Asst. Professor, ECE Dept., CMRCET, Hyderabad, AP-India, suresh1516@gmail.com
[2] Project Manager, Centre for Integrated Solutions, Secunderabad, AP-India

***Abstract:* Image compression is an extended research area from a long time. Various compression schemes were developed for the compression of image in gray scaling and color image compression. Basically the compression is focused for lossy or lossless color image compression based on the need of applications. Where lossy compression are higher in compression ratio but are observed to be lower in retrieval accuracy. Various compression techniques such as JPEG and JPEG-2K architectures are developed to realize such and method. But with the need in high accuracy retreivation these techniques[8] are not suitable. To achieve nearby lossless compression[1] various other coding methods were suggested like lifting scheme coding. This coding result in very high retrieval accuracy but gives low compression ratio. This limitation is a bottleneck in current image compression architectures. So, there is a need in the development of a compressing approach where both higher compression as well as higher retrieval accuracy is obtained.**

***Keyword:* Image Compression, Context Modeling, Arial Images, PSNR, Compression Ratio**

I. INTRODUCTION

Digital imagery has had an enormous impact on industrial, scientific and computer applications. Image coding has been a subject of great commercial interest in today's world. Uncompressed digital images require considerable storage capacity and transmission bandwidth. Efficient image compression solutions are becoming more critical with the recent growth of data intensive, multimedia-based web applications. An image is a positive function on a plane. The value of this function at each point specifies the luminance or brightness of the picture at that point. Digital images are sample versions of such functions, where the value of the function is specified only at discrete locations on the image plane, known as pixels. During transmission of these pixels the pixel data must be compressed to match the bit rate of the network.

In order to be useful, a compression algorithm has a corresponding decompression algorithm[3] that, given the compressed file, reproduces the original file. There have been many types of compression algorithms developed. These algorithms fall into two broad types, loss less algorithms and lossy algorithms. A lossless algorithm reproduces the original exactly. A lossy algorithm, as its name implies, loses some data. Data loss may be unacceptable in many applications. For example, text compression must be lossless because a very small difference can result in statements with totally different meanings. While processing with documented images such as digitized map images the compression required are to be totally lossless, as a minor variation in the image data may result in a wrong representation of the map information in the case of roads, vegetation, dwelling etc. Various research works were carried out on both lossy and lossless image compression[1] in past. The image compression committee has come out with the JPEG committee with a release of a new image-coding standard, JPEG 2000 that serves the enhancement to the existing JPEG system. The JPEG 2000 implements a new way of compressing images based on the wavelet transforms in contrast to the transformations used in JPEG standard. There is a majority of today's Internet bandwidth is estimated to be used for images and video transmission. Recent multimedia applications for handheld and portable devices place a limit on the available wireless bandwidth.

II. STATISTICAL IMAGE CODING

A binary image can be considered as a message, generated by an information source. The idea of statistical modeling is to describe the message symbols (pixels) according to the probability distribution of the source alphabet (binary alphabet, in our case). Shannon has shown that the information content of a single symbol (pixel) in the message (image) can be measured by its entropy:

$$H_{pixel} = -Log_2 P,$$

Where P is the probability of the pixel. Entropy of the entire image can be calculated as the average entropy of all pixels:

$$H_{image = -1/n} \sum^{n}{}_{i=1} Log_2 P_{i},$$

Where Pi is the probability of ith pixel and n is the total number of pixels in the image.
If the probability distribution of the source alphabet (black and white pixels) is a priori known, the entropy of the probability model can thus be expressed as:

$$H = -P_w Log_2 P_w - P_B Log_2 P_B \ ,$$

Where P_w and P_B are the probabilities of the white and black pixels, respectively. The more sophisticated Bayesian sequential estimator calculates probability of the pixel on the basis of the observed pixel frequencies as follows:

© 2011 ACEEE
DOI: 01.IJSIP.02.01.532

$$p(x^t)=\begin{cases} p_W^t=\dfrac{n_W^t+\delta}{n_W^t+n_B^t+2\delta}, & \text{if } x^t \text{ is } \textit{white} \\ p_B^t=1-p_W^t, & \text{if } x^t \text{ is } \textit{black} \end{cases}$$

Where n^t_W, n^t_B are the time-dependent counters, p^t_W, p^t_B are the probabilities for white and black colors respectively, and δ is a constant. Counters n^t_W and n^t_B start from zero and are updated after the pixel has been coded (decoded). As in [JBIGl], we use δ = 0.45. The cumulative equation for entropy is used to estimate the average bit rate and calculate the ideal code length.

III. CONTEXT BASED STATISTICAL MODEL

The pixels in an image form geometrical structures with appropriate spatial dependencies.[2] The dependencies can be localized to a limited neighborhood, and described by a context-based statistical model. In this model, the pixel probability is conditioned on the context C, which is defined as distinct black-white configuration of neighboring pixels within the local template. For binary images, the pixel probability is calculated by counting the number of black (n^C_B) and white (n^C_W) pixels appeared in that context in the entire image:

$$p(x)=\begin{cases} p_W^C=\dfrac{n_W^C}{n_W^C+n_B^C}, & \text{if } x=\textit{white} \\ p_B^C=1-p_W^C, & \text{if } x=\textit{black} \end{cases},$$

Here, p^C_B and p^C_W are the corresponding probabilities of the black and white pixels. The entropy H(C) of a context C is defined as the average entropy of all pixels within the context:

$$H(C) = - p_W{}^C{}_{\log 2}\, p_W{}^C - p_B{}^C{}_{\log 2}\, p_B{}^C$$

A context with skew probability distribution has smaller entropy and therefore smaller information content. The entropy of an N-level context model is the weighted sum of the entropies of individual contexts:

$$H_N = - \sum\nolimits^{N}_{j=1} p(C_j).\ (p_W{}^{Cj}.\ \log_2 p_W{}^{Cj} + p_B{}^{Cj}.\log_2 p_B{}^{Cj})$$

In principle, a skewed distribution can be obtained through conditioning of larger regions by using larger context templates. However, this implies a larger number of parameters of the statistical model and, in this way, increases the model cost, which could offset the entropy savings. Another consequence is the "context dilution" problem occurring when the count statistics are distributed over too many contexts, thus affecting the accuracy of the probability estimates.

IV. ALGORITHM

The context size is a trade-off between the prediction accuracy and learning cost (in dynamic modeling) or model overhead (in semi-adaptive modeling). A larger template size gives us a theoretically better pixel prediction. This results in a skewer probability distribution and lower bit-rates.

CONSTRUCTION PROCEDURE

To construct a context tree, the image is processed and the statistics n^t_W and n^t_B are calculated for every context in the full tree, including the internal nodes. The tree is then pruned by comparing the children and parents nodes at each level. If compression gain is not achieved from using the children nodes instead of their parent node, the children are removed from the tree and their parent will become a leaf node. The compression gain is calculated as:

Gain (C, C_W, C_B)= l(C) - l(C_W) - l(C_B) – Split Cost ,

Where C is the parent context and C_W and C_B are the two children nodes. The code length *l* denotes the total number of output bits from the pixels coded using the context. The cost of storing the tree is integrated into the Split Cost parameter. The code length can be calculated by summing up the entropy estimates of the pixels as they occur in the image: $l(C) = \sum_t \log p^t(C)$ The probability of the pixel is calculated on the basis of the observed frequencies using a Bayesian sequential estimator:

$$p^t(C)=\begin{cases} p_W^t(C)=\dfrac{n_W^t(C)+\delta}{n_W^t(C)+n_B^t(C)+2\delta}, & \text{if } t^{\text{th}} \text{ pixel is } \textit{white} \\ p_B^t(C)=1-p_W^t(C), & \text{if } t^{\text{th}} \text{ pixel is } \textit{black} \end{cases}$$

Where n^t_W , n^t_B are the time-dependent frequencies, and p^t_W, p^t_B are the probabilities for white and black colors respectively, and δ = 0.45, as in [JBIG1].

The template form and the pixel order in this example are optimized for topographic images.

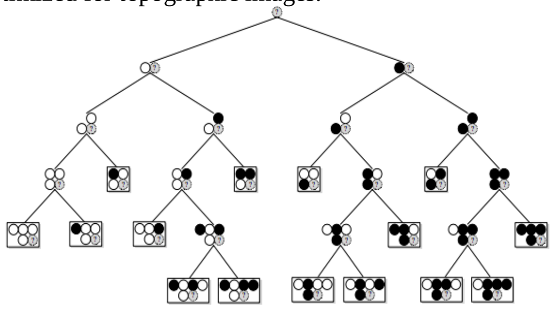


Figure: 1. Illustration of a context tree.

In the bottom-up approach, the tree is analyzed from the leaves to the root. A full tree of k_{Max} levels is first constructed by calculating statistics for all contexts in the tree. The tree is then recursively pruned up to level k_{Min}, using the same criterion as in the top-down approach. The gain is calculated using the code length equation using *l*(C). The code lengths from the children contexts *l*(C_W) and *l*(C_B) are derived from the previous level of the recursion. The sub-trees of the nodes that do not deliver positive compression gain are removed from the tree. A sketch of the implementation is shown in Figure and the algorithm is illustrated in Figure.

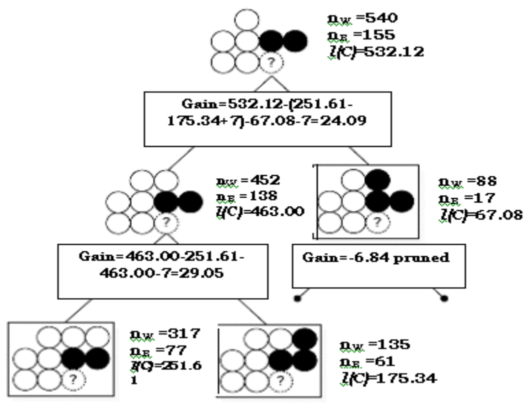
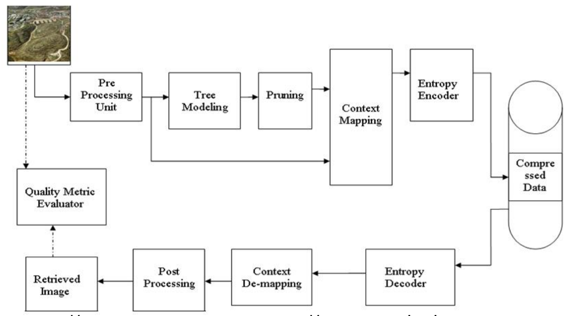


Figure: 2. Illustration of bottom-up tree pruning.

The bottom-up approach can be implemented using only one pass over the whole image. Unfortunately, high k_{MAX} values will result in huge memory consumption. For this reason, a two-stage bottom-up pruning procedure was proposed in. In the first stage, the tree is constructed from the root to level K_{START} and then recursively pruned until level $k_{Min.}$ In the second stage, the remaining leaf nodes at the level K_{START} are expanded up to level k_{Max} and then pruned until level K_{START} ' In this way, the memory consumption depends mainly on the choice of the K_{START} because only a small proportion of the nodes at that level remains after the first pruning stage. The starting level K_{START} is chosen as large as the memory resources permit. This approach of modeling & pruning results in a higher compression of pixel representation in given image sample.

V. SYSTEM DESIGN

The suggested context modeling for color image compression is developed for the evaluation of Arial map images. These images are higher in textural variation with high color contrast. A compression scheme for such an image is designed and the block diagram for this method is as shown below,



This designed system read the color image information and transform to gray plane with proper data type conversion for the computation. During pre processing operation the map image is equalized with histogram equalization to normalize the intensity distribution to be in uniform level. This equalized image is then processed with binarization by using thresholding technique to estimate the required region out of the whole. A mathematical morphology[10] operation is apply to extract the bounding regions and curves in the map image.

The extracted regions are then passed to context tree modeling[7] for obtaining the context feature of the obtained regions. A tree model is developed as explained in previous sections for the obtained context features. The contexed features are quite large in count and are required to be reduced for higher compression.

To achieve lower context counts a pruning operation is performed. A Pruning is a hierarchical coding technique for dimensionality reduction with a tracing of obtained context features in a hierarchical tree manner with branches and leaf projecting towards high dominative context features comparative to lower context features discarding at intermediate level. This pruning process hence ends out at minimum number of dominative context features resulting in low context information for context mapping. This results in faster computation of image data mapping with context feature for entropy encoding.

Context mapping is a process of transforming the image coefficient to a context tree model mapping[9] with reference to obtained pruning output. The image pixels are mapped with pruning output and a binary stream indicating the mapping information is generated. This stream is passed to entropy encoder for performing binary compression using Huffman entropy coding.[4] [14]

The dequantized information is passed to the demapping operation. The image coefficients are retrieved with a demapping of the context information to the dequantized data to obtain the original image information back. The demapping operation is carried out in a similar fashion as like the mapping operation with the reference of context table. For the regeneration of pixel coefficient the same context tree is used for the reverse tree generated to retrieve pixel coefficient. The obtained coefficients are post processed for the final generation of the image.

This unit realigns the pixel coefficient to the grid level based on the generated context index during pre processing. The image retrieved after the computation is evaluated for PSNR value in the evaluator unit for the performance evaluation of suggested architecture with estimation accuracy as the quality factor.

DOI: 01.IJSIP.02.01.532

VI. RESULTS

For the processing of a developed system Arial captured map images with color information is passed. The images are passed in a higher resolution to attain best context extraction so as to maintain highest accuracy. The images are taken in a JPEG format and passed to the processing unit to compute histogram diagram for the normalization and further processing.

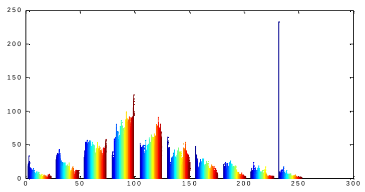


Figure: 5. Histogram plot for the given query Image

Figure: **6.**Histogram Equalized Image

Figure: 7.Binarized Image

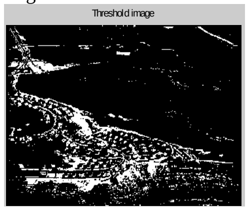


Figure: 8. Threshold Image

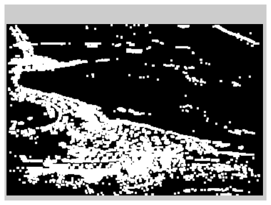


Figure: 9. Extracted Image

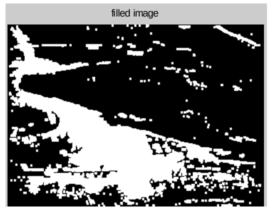


Figure: 10. Filled Image

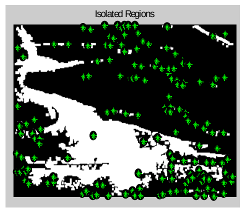


Figure: 11. Context Features

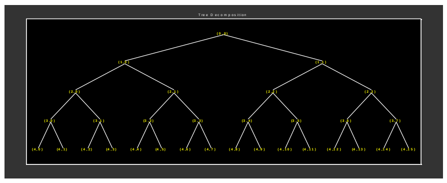
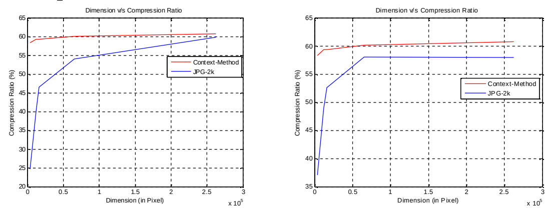


Figure: 12. The Hierarchical Tree Modeling

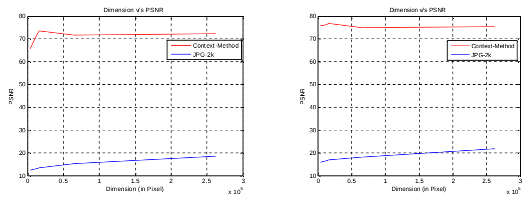
PERFORMANCE OBSERVATIONS

A performance evaluation is given for calculation of retrieval accuracy and compression ratio by evaluation of PSNR and percentage of compression ratio for the given query samples.

Performance Observation for 8 & 24-Bit Depth of Sample-1



In 8- bit depth graph at low bit level compression is high for the context tree method because it is a hierarchical context tree where as existing JPEG2000 has very low compression but for larger data JPEG2000 almost tried to reach context method. In 24-bit depth also gives the same information.



In 8- bit depth graph at low bit level PSNR value for the context tree method is in between 65 to 75 % where as existing JPEG2000 has very low PSNR value in between 10 to 20 %, but for 24-bit depth JPEG2000 is reached to 20 to 25 %.

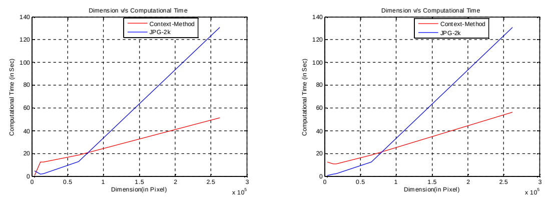


Figure: 13. Plots of Context Method and JPEG2000

In 8- bit depth graph at low bit level computational time for the context tree method is high compare to JPEG2000 has very low computational time, but for larger data JPEG2000 has taken very high computational time where as context tree has taken low computational time.

DOI: 01.IJSIP.02.01.532

OBSERVATION TABLES

Image Bit Depth: 8 - Bit Depth

Sample	8- Bit Depth	Context Method			JPEG-2000		
	Dimension	Compression Ratio	PSNR	Computational Time	Compression Ratio	PSNR	Computational Time
S1	64*64	58.33	65.81	0.70	24.82	12.39	0.73
	110*110	59.34	71.35	12.5	40.04	13.07	1.78
	128*128	59.34	73.58	12.5	46.51	13.32	2.37
	256*256	60.11	71.80	18.7	54.06	15.22	12.8
	512*512	60.74	72.27	51.5	59.77	18.51	130.9
S2	64*64	58.33	65.81	9.30	24.82	12.39	0.73
	110*110	59.34	71.35	9.30	40.04	13.07	1.92
	128*128	59.34	73.58	10.9	46.51	13.32	2.40
	256*256	60.11	71.80	18.7	54.06	15.22	12.8
	512*512	60.74	72.27	54.6	59.77	18.51	129.7
S3	64*64	58.33	65.81	9.30	24.82	12.39	0.73
	110*110	59.34	71.35	9.30	40.04	13.07	1.89
	128*128	59.34	73.58	12.5	46.51	13.32	2.31
	256*256	60.11	71.80	18.7	54.06	15.22	13.01
	512*512	60.74	72.27	57.8	59.77	18.51	133.6

Image Bit Depth: 24 - Bit Depth

Sample	24- Bit Depth	Context Method			JPEG-2000		
	Dimension	Compression Ratio	PSNR	Computational Time	Compression Ratio	PSNR	Computational Time
S1	64*64	58.33	75.80	12.5	37.03	15.90	0.671
	110*110	59.34	76.19	10.9	48.93	16.49	1.73
	128*128	59.34	76.85	10.9	52.57	16.89	2.23
	256*256	60.11	74.92	1.87	58.07	18.13	12.4
	512*512	60.74	75.45	56.2	57.98	21.75	130.6
S2	64*64	58.33	75.80	9.30	37.03	15.97	0.671
	110*110	59.34	76.19	12.0	48.93	16.49	1.72
	128*128	59.34	76.85	12.5	52.57	16.89	2.34
	256*256	60.11	74.92	23.4	58.07	18.13	12.46
	512*512	60.74	75.45	56.2	57.98	21.75	130.9
S3	64*64	58.33	75.80	7.80	37.03	15.97	0.75
	110*110	59.34	76.19	12.5	48.93	16.49	1.78
	128*128	59.34	76.85	14.0	52.57	16.89	2.29
	256*256	60.11	74.92	21.8	58.07	18.13	13.26
	512*512	60.74	75.45	56.2	57.98	21.75	131.2

VII. CONCLUSION

This Paper presents a compression approach in image processing applications using a hierarchical context modeling of highly varying color images with practical information of Arial mapping. The developed context tree modeling is focused with the objective of attaining minimum error and faster computations in processing these mapping images for real time applications. For the developed system the quality metric of situation accuracy with respect to PSNR value is computed and observed to be a higher value giving suggested method as feasible solutions for fast, high and lossless compression in practical environments.

VIII. REFERENCES

[1] Alexander Akimov, Alexander Kolesnikov, and Pasi Fränti, "Lossless Compression of Color Map Images by Context Tree Modeling", IEEE Transactions On Image Processing, Vol. 16, NO. 1, January 2007.

[2] Samet H. Applications of Spatial Data Structures: Computer Graphics,Image Processing, GIS. MA: Addison-Wesley, Reading. May 2006

[3] Pajarola R, Widmayer P. Spatial indexing into compressed raster images: how to answer range queries without decompression. Proc. Int. Workshop on Multimedia DBMS (Blue Mountain Lake, NY, USA), 94-100. 1, May 2000

[4] Hunter R., Robinson A.H. International digital facsimile coding standards. Proc. of IEEE, 68 (7), 854-867. 2002

[5] Urban S.J. Review of standards for electronic imaging for facsimile systems. Journal of Electronic Imaging, 1(1): 5-21. 6,May 2008

[6] Salomon D. Data compression: the complete reference. New York: Springer- Verlag.] Arps RB., Truong T.K. Comparison of international standards for lossless still image compression. Proceedings of the IEEE 82: 889-899. 2003

[7] Rissanen J.J., Langdon G.G.) Universal modeling and coding. IEEE Trans. Inform. TheoryIT-27: 12-23. 2000

[8] Netravali A.N., Mounts F.W. Ordering Techniques for Facsimile Coding: A Review. Proceedings of the IEEE, 68 (7): 796-807. 2003

[9] Capon J. A probabilistic model for run-length coding of pictures. IRE Tran. Information Theory, IT-5: 157-163. 12, March 2007

[10] Shannon c.E. A mathematical theory of communication. Bell System. Tech Journal 27: 398-403. 1, Nov 2001

[11] Vitter J. Design and Analysis of dynamic Huffman codes. Journal of Association for Computing Machinery, 34:825-845. 5, May 2000

[12] Rice RF. Some practical universal noiseless coding techniques. Proc. Data Compression Conference (Snow Bird, Utah, USA), 351360.

[13] Colombo S.W. Run-length encoding. IEEE Trans. Inform Theory, IT-12: 399-401. 4, Aug 2005

[14] Huffman D.A. A method for the construction of minimum redundancy codes. Proc. of IEEE, 40 (9): 1098-110 I. 2007

[15] Shannon c. E. A mathematical theory of communication. Bell Syst. Tech Journal 27: 398-403. 5, May 2000

[16] Wao Y. Wu Y. J.-M. Vector Run-Length Coding of Bi-Level Images. Proceedings Data Compression Conference, Snowbird, Utah, USA, 289-298. 5, May 2000

[17]ITU-T (CCITT) Recommendation TA. Kunt M., Johnsen O. Block Coding: A Tutorial Review. Proceedings of the IEEE, 68 (7): 770-786. 5, May 2000

[18] Franti P., Nevalainen O. Compression of binary images by composite methods basically on the block coding. Journal of Visual Communication, Image Representation 6 (4): 366-377. 26, Jun 1999

[19] Rissanen J.J., Langdon G.G. Arithmetic coding. IBM Journal of Research, Development 23: 146-162. 2007

[20] Langdon G.G., Rissanen J. Compression of black-white images with arithmetic coding. IEEE Trans. Communications 29(6):858-867.2000